

Sprawozdanie WMM Laboratorium 4

Miłosz Andryszczuk 331355

Grupa 104 Piątek 14:15

numer_obrazu = numer_indeksu % liczba_obrazow = 331355 % 36 = **11**

Zadanie 1

Filtracji barwnego obrazu cyfrowego

Obrazy zaszumione szumem Gaussowskim i impulsowym poddano przetwarzaniu filtrem wygładzającym (Gaussa) i medianowym. Dla każdego filtra przetestowano różne rozmiary masek: 3x3, 5x5, 7x7.

Tabela wartości PSNR [dB] dla poszczególnych obrazów wynikowych:

Rodzaj szumu	Rodzaj filtru	Maska 3x3	Maska 5x5	Maska 7x7
Gaussowski	Gaussa	26.04	25.10	24.19
	Medianowy	23.92	24.65	23.13
Impulsowy	Gaussa	25.07	24.62	23.94
	Medianowy	24.13	24.72	23.13

Obraz oryginalny:



Obraz z szumem Gaussowskim:



Obraz z szumem impulsowym:



Obrazy wynikowe:

1. Szum Gaussowski

1.1. Filtr Gaussa

Oryginał



Maska 3x3 26.04dB



Maska 5x5 25.10dB



Maska 7x7 24.19dB



1.2. Filtr medianowy

Oryginał



Maska 3x3 23.92dB



Maska 5x5 24.65dB



Maska 7x7 23.13dB



2. Szum impulsowy

2.1. Filtr Gaussa

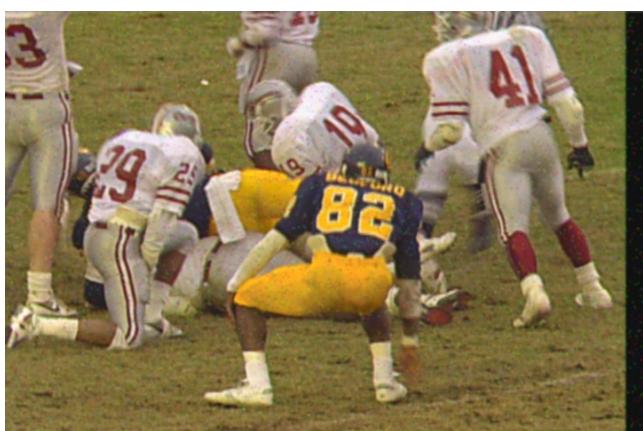
Oryginał



Maska 3x3 25.07dB



Maska 5x5 24.62dB



Maska 7x7 23.94dB



2.2. Filtr medianowy

Oryginał



Maska 3x3 24.13dB



Maska 5x5 24.72dB



Maska 7x7 23.13dB



Wnioski

1. Wpływ rozmiaru maski na skuteczność filtracji i zniekształcenie obrazu

Filtr Gaussa lepiej radzi sobie z tłumieniem szumu dla większej maski. Widać to w szczególności dla szumu impulsowego, dla którego liczba punktów odstających od obrazu jest znacznie mniejsza dla większego rozmiaru maski (zmniesza się efekt soli). Obraz jednak staje się mniej ostry, a krawędzie obiektów rozmyte. Dla szumu gaussowskiego filtr Gaussa radzi sobie najlepiej dla jak najmniejszej maski (3x3). Większe rozmiary powodują znaczny spadek ostrości obrazu.

Filtr medianowy bardzo dobrze radzi sobie z szumem impulsowym już dla maski 3x3, a coraz większe rozmiary maski powodują tylko wygładzenie szczegółów i obraz staje się nienaturalnie „płaski”. Podobnie jest dla szumu gaussowskiego – większa maska skutkuje lepszą filtracją, ale gorszą jakością.

Z reguły większa maska skutkuje lepszymi efektami filtracji szumów, jednak powoduje to spadek ostrości, wygładzenie detali i ogólne pogorszenie jakości obrazu.

2. PSNR a obiektywna ocena obrazów

Miara PSNR ogólnie dobrze ocenia jakość obrazu i jego podobieństwo do oryginału. W każdym przypadku dla maski 7x7 obraz wygląda najgorzej i faktycznie PSNR jest najmniejsze. Jednak dla niektórych przypadków wskaźnik ten nie zgadza się z obiektywną opinią, szczególnie dla obrazów z szumem impulsowym. Dla takiego obrazu filtr medianowy najlepiej poradził sobie z maską 3x3, a mimo to wskaźnik PSNR jest najwyższy dla obrazu odszumionego maską 5x5. Polemizować można również, czy filtr Gaussa faktycznie lepiej działa z maską 3x3, a nie 5x5 dla szumu impulsowego (a tak wskazuje PSNR).

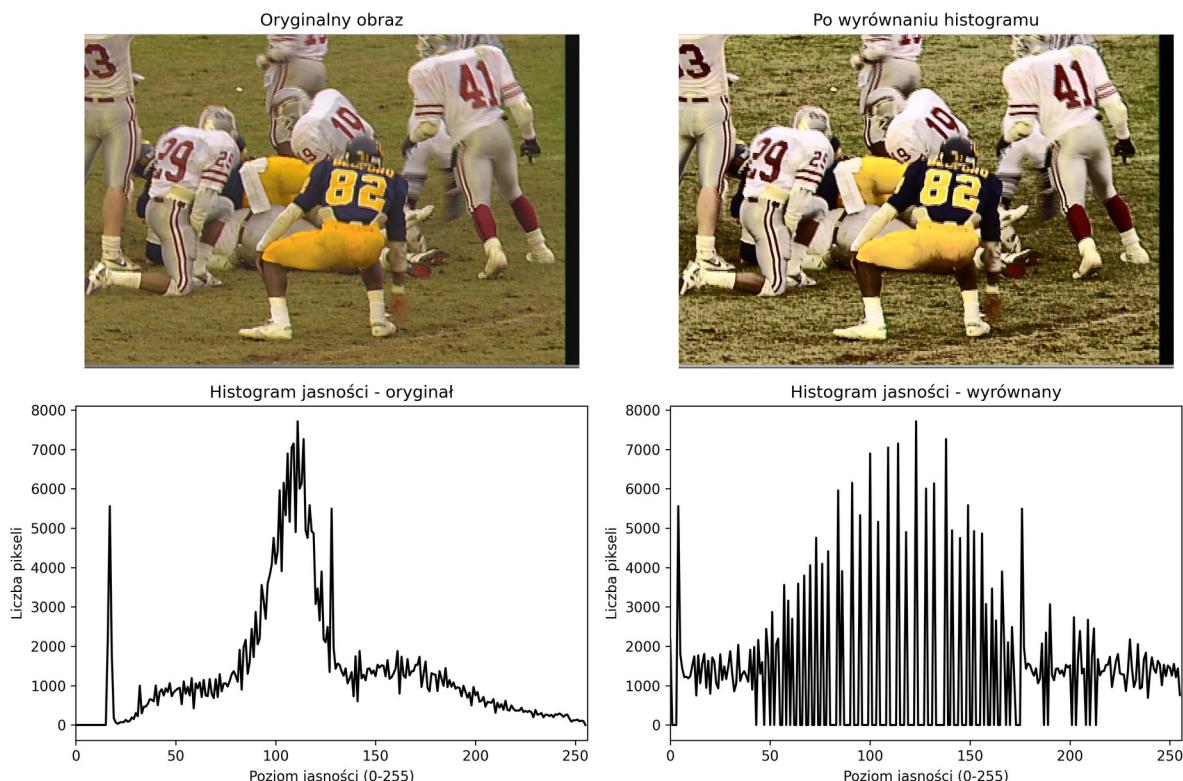
PSNR jest dobrym wskaźnikiem jakości danego obrazu, ponieważ bardzo dobrze odróżnia obrazy o skrajnie różnej jakości. Natomiast w niektórych przypadkach obrazy o niższym PSNR okazują się być lepsze. Wszystko zależy jednak od tego, na czym nam zależy i na co zwracamy uwagę – jak najlepsze usunięcie szumów kosztem jakości, czy może zachowanie ostrości i detali oryginalnego obrazu.

Zadanie 2

Wyrównanie histogramu dla obrazu barwnego

Wczytany plik został przekonwerterowany z formatu RGB do YCbCr. Następnie dokonano wyrównania histogramu dla kanału Y odpowiadającego za jasność obrazu i powrócono do formatu RGB.

Sporządzone zostały również wykresy histogramu obu obrazów. Wskazują one ile pikseli ma określoną jasność w skali od 0 do 255.



Histogram jasności przed wyrównaniem wskazuje na silne skupienie wartości w wąskim zakresie środkowej jasności, co oznacza, że obraz pierwotnie był lekko przygaszony, bez dużych różnic między jasnymi a ciemnymi partiami. Po wyrównaniu histogramu rozkład poziomów jasności stał się znacznie bardziej rozciągnięty na całym zakresie od 0 do 255.

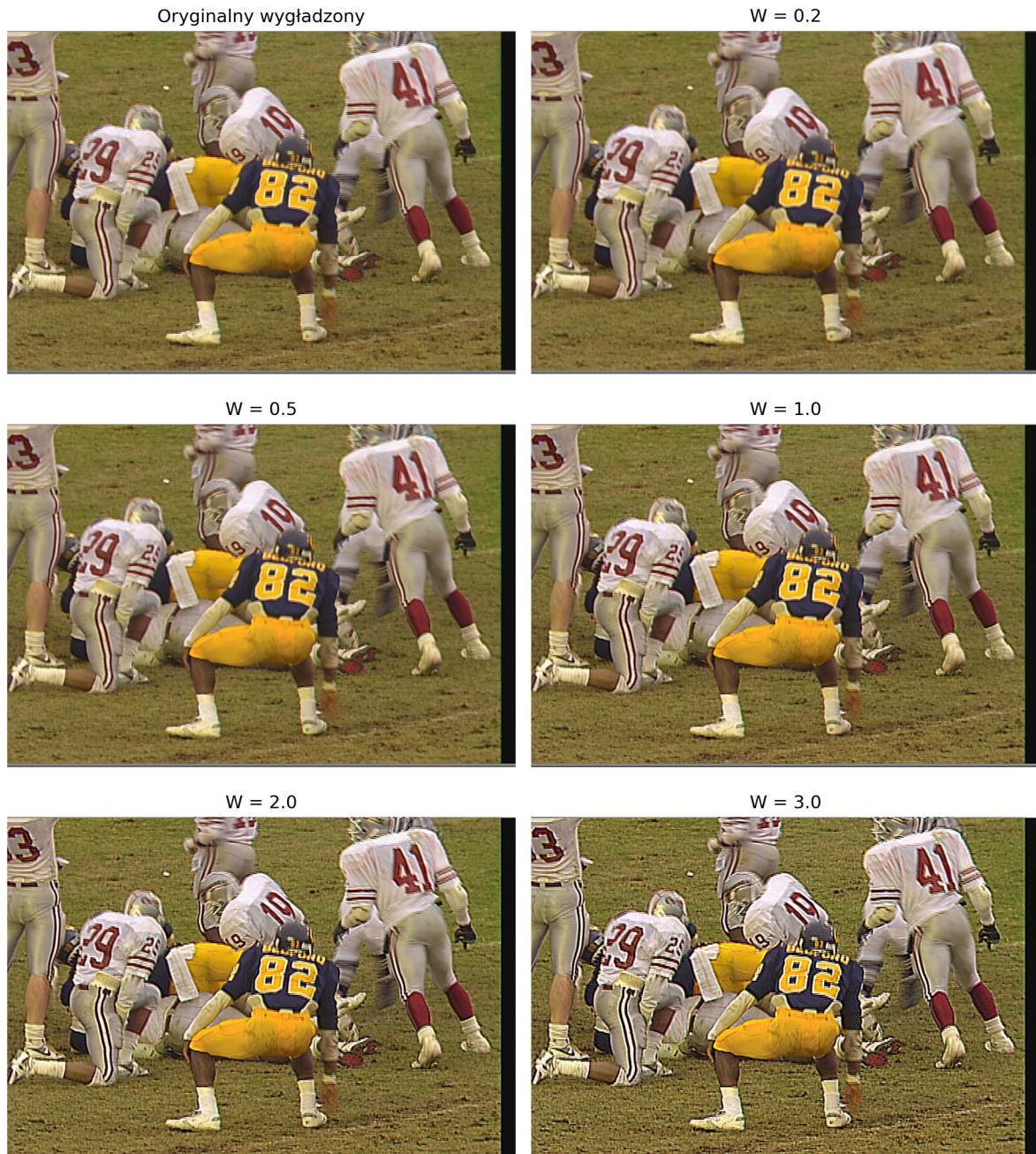
Obraz po wyrównaniu histogramu stał się wyraźnie bardziej kontrastowy. Jasne obszary, takie jak białe koszulki zawodników oraz ciemne fragmenty boiska i cieni, nabrali większej głębi, co sprawia, że scena wydaje się „ostrzejsza”. Jednak jednocześnie pojawił się efekt przerysowania barw, które są zbyt nienaturalne. Subiektywnie można uznać go za lepszy pod względem widoczności szczegółów, lecz mniej realistyczny kolorystycznie.

Zadanie 3

Wystrzenie obrazu za pomocą filtra Laplace'a

Obraz oryginalny wstępnie wygładzono filtrem Gaussa, aby ewentualne szумy nie zostały uwydartnione. Następnie obliczono składowe wysokoczęstotliwościowe korzystając z filtru Laplace'a i przemnożone przez wagę dodano je do oryginalnego obrazu.

Porównanie oryginalnego wygładzonego obrazu i obrazów wyostrzonych z wagami W:





Wnioski

Na podstawie zaprezentowanych wyników można zauważyc, że wzrost wartości wagi składowej wysokoczęstotliwościowej wpływa bezpośrednio na stopień wyostrzenia obrazu. Dla niskich wartości wagi ($W = 0.2$) efekt wyostrzania jest bardzo subtelny i niemal niezauważalny. Wraz ze wzrostem wagi obraz staje się nieco wyraźniejszy, kontury są lepiej zdefiniowane, a szczegóły delikatnie podkreślone. W przypadku wag $W = 1.0$ lub 2.0 obraz nadal zachowuje naturalny wygląd, jednak wyższe wartości prowadzą do dalszego wzmacniania krawędzi, przez co obraz zaczyna wyglądać nienaturalnie. Dla wag, takich jak $W = 5.0$ czy $W = 10.0$, krawędzie są nadmiernie zaakcentowane, a cały obraz zyskuje przerysowany charakter, z dużym kontrastem (widać to szczególnie dla trawy w tle).

Wnioskując, wpływ wagi składowej wysokoczęstotliwościowej polega na tym, że im większa waga, tym silniejsze wyostrzenie i bardziej agresywne wzmacnienie krawędzi, ale jednocześnie rośnie ryzyko sztuczności i spadku jakości. Najlepsze, przyjemne dla oka efekty uzyskuje się dla wartości wagi w przedziale od około 0.5 do 1.0, ponieważ zapewniają one dobrą równowagę między wyraźnością a naturalnością obrazu.

Kod źródłowy

Na kolejnych stronach przedstawiono kod do poszczególnych zadań.

Zadanie 1

```
import cv2
import numpy as np

def calcPSNR(img1, img2):
    imax = 255.0 ** 2
    mse = ((img1.astype(np.float64) - img2) ** 2).sum() / img1.size
    return 10.0 * np.log10(imax / mse)

original = cv2.imread("original.png")
impulse_noise = cv2.imread("impulse.png")
gauss_noise = cv2.imread("gaussian.png")

kernel_sizes = [(3, 3), (5, 5), (7, 7)]

for ksize in kernel_sizes:
    print(f"\nMaska: {ksize}")
    save_dir = f"kernel_{ksize[0]}"

    # Szum gaussowski
    blur_gauss = cv2.GaussianBlur(gauss_noise, ksize, 0)
    median_gauss = cv2.medianBlur(gauss_noise, ksize[0])

    cv2.imwrite(f"{save_dir}/gauss_noise/gauss.png", blur_gauss)
    cv2.imwrite(f"{save_dir}/gauss_noise/median.png", median_gauss)

    psnr_blur_gauss = calcPSNR(original, blur_gauss)
    psnr_median_gauss = calcPSNR(original, median_gauss)

    print("szum gaussowski:")
    print(f"Gauss blur: {psnr_blur_gauss:.2f} dB")
    print(f"Median blur: {psnr_median_gauss:.2f} dB\n")

    # Szum impulsowy
    blur_impulse = cv2.GaussianBlur(impulse_noise, ksize, 0)
    median_impulse = cv2.medianBlur(impulse_noise, ksize[0])

    cv2.imwrite(f"{save_dir}/impulse_noise/gauss.png", blur_impulse)
    cv2.imwrite(f"{save_dir}/impulse_noise/median.png", median_impulse)

    psnr_blur_impulse = calcPSNR(original, blur_impulse)
    psnr_median_impulse = calcPSNR(original, median_impulse)

    print("szum impulsowy:")
    print(f"Gauss blur: {psnr_blur_impulse:.2f} dB")
    print(f"Median blur: {psnr_median_impulse:.2f} dB")

    # Wyświetlenie obrazów
    cv2.imshow("Original", original)
    cv2.imshow("Impulse noise", impulse_noise)
    cv2.imshow("Gauss noise", gauss_noise)

    cv2.imshow("Gauss noise, Gauss blur", blur_gauss)
    cv2.imshow("Gauss noise, Median blur", median_gauss)

    cv2.imshow("Impulse noise, Gauss blur", blur_impulse)
    cv2.imshow("Impulse noise, Median blur", median_impulse)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Zadanie 2

```
import cv2
import matplotlib.pyplot as plt

def plot_histogram(channel, title, color='black'):
    hist = cv2.calcHist([channel], [0], None, [256], [0, 256])
    plt.plot(hist, color=color)
    plt.title(title)
    plt.xlabel("Poziom jasności (0-255)")
    plt.ylabel("Liczba pikseli")
    plt.xlim([0, 256])

image = cv2.imread('original.png')
image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

# Konwersja do przestrzeni YCrCb
image_ycrcb = cv2.cvtColor(image, cv2.COLOR_BGR2YCrCb)
y, cr, cb = cv2.split(image_ycrcb)

# Wyrównanie histogramu jasności
y_eq = cv2.equalizeHist(y)

# Konwersja z powrotem do RGB
image_eq_ycrcb = cv2.merge((y_eq, cr, cb))
image_eq = cv2.cvtColor(image_eq_ycrcb, cv2.COLOR_YCrCb2RGB)

cv2.imwrite("image_equalized.png", cv2.cvtColor(image_eq, cv2.COLOR_RGB2BGR))

fig, axs = plt.subplots(2, 2, figsize=(12, 8))

# Obraz oryginalny
axs[0, 0].imshow(image_rgb)
axs[0, 0].set_title("Oryginalny obraz")
axs[0, 0].axis('off')

# Obraz po wyrównaniu histogramu
axs[0, 1].imshow(image_eq)
axs[0, 1].set_title("Po wyrównaniu histogramu")
axs[0, 1].axis('off')

# Histogram jasności oryginału
plt.subplot(2, 2, 3)
plot_histogram(y, "Histogram jasności - oryginal")

# Histogram jasności po wyrównaniu
plt.subplot(2, 2, 4)
plot_histogram(y_eq, "Histogram jasności - wyrównany")

plt.tight_layout()
plt.savefig("histogram_comparison.png", dpi=300)
plt.show()
```

Zadanie 3

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

def sharpen_laplacian(img_f, weight):
    laplace = cv2.Laplacian(img_f, cv2.CV_32F)

    laplace_vis = cv2.convertScaleAbs(laplace)
    cv2.imwrite("laplace.png", laplace_vis)

    result = cv2.addWeighted(src1=img_f, alpha=1.0, src2=laplace, beta=-weight, gamma=0)
    return np.clip(result, 0, 255).astype(np.uint8)

img = cv2.imread("original.png")
img_rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

blurred = cv2.GaussianBlur(img_rgb, (3, 3), 0)

blurred_f = blurred.astype(np.float32)

weights = [0.2, 0.5, 1.0, 2.0, 3.0, 5.0, 10.0]

fig, axs = plt.subplots(4, 2, figsize=(10, 15))

# Obraz oryginalny
axs[0, 0].imshow(img_rgb)
axs[0, 0].set_title("Oryginalny wygładzony")
axs[0, 0].axis('off')

# Wyostrzanie dla każdej wagi
for i, w in enumerate(weights):
    row = (i + 1) // 2
    col = (i + 1) % 2
    sharpened = sharpen_laplacian(blurred_f, w)
    axs[row, col].imshow(sharpened)
    axs[row, col].set_title(f"W = {w}")
    axs[row, col].axis('off')

plt.tight_layout()
plt.savefig("wyostrzanie_laplace.png", dpi=300)
plt.show()
```